ERIC Document Resume | See Instruction Sheet | DATE OF RESUME 7/7/65 | DATE RECEIVED

A. PUBLICATION DATA

1. ACCESSION NUMBER: ED000929
2. ORIGINATOR CONTROL NO(S).: 3580
3. IDENTIFICATION NO.:

DO NOT WRITE IN SPACE BELOW

4. AUTHOR(S):
Mager, Robert F.

5. TITLE OF DOCUMENT ABSTRACTED:
Getting Started With Programed Instruction

6. PUBLICATION TITLE:
N.A.

7. EDITOR(S):
N.A.

8. PLACE OF ORIGIN
ORG. ASEE
ST. Univ. of Illinois
CITY, STATE Urbana, Illinois
TELE. EXT.
INST/LOC CODE

9. PUBLISHER OR SOURCE
The American Society for Engineering Education

10. VOL.	11. PAGES	12. MO./YR	13. TOTAL PP.
N.A.	N.A.	Nov. 62	13

MICROFILM COPY AVAILABLE
YES ☒ NO ☐

COPYRIGHT GRANTED
YES ☐ NO ☒

COPYRIGHT RELEASE FILED
YES ☐ NO ☐

TYPE OF REPORT
PROPOSAL ☐
IN-PROGRESS ☐
FINAL ☒
FOLLOW-UP ☐

OTHER ERIC REFERENCES:

ERIC SATELLITE CODE

B. INDEXING DATA

14. SUBJECT AREA

15. TYPE OF ENDEAVOR
report

16. INDEXING TERMS (6-12)
teaching methods Y-25
teaching aids Y-22
programmed learning P-20
programmed texts P-21
materials 0-10

17. ABSTRACT (MAX. 250 WORDS)

A brief article prepared for the Committee on Programed Instruction of the American Society for Engineering Education. Offers suggestions for writing programs for presentation as programed instruction as well as suggestions regarding the use of programed instruction materials. The pamphlet is designed to help instructors evaluate the usefulness of the suggested techniques in their own instructional situations.

A selected annotated bibliography is appended.

ERIC No.

18. SOURCE OF ABSTRACT

C. FUNDING DATA

19. SUPPORTING ORGANIZATION

20. DIVISION OR DEPT.

21. CONTRACT NO.
PROJECT NO.

22. TOTAL COST OF PROJECT

23. DURATION OF PROJECT

U.S. Department of Health, Education, and Welfare
Office of Education

ERIC Form 1

R-7-23

U. S. DEPARTMENT OF HEALTH, EDUCATION AND WELFARE
Office of Education

This document has been reproduced exactly as received from the person or organization originating it. Points of view or opinions stated do not necessarily represent official Office of Education position or policy.

8570

0320

GETTING STARTED WITH PROGRAMMED INSTRUCTION

prepared by

Robert F. Mager

26 November 1962

for the

Committee on Programmed Instruction

of

THE AMERICAN SOCIETY FOR ENGINEERING EDUCATION

(This Committee is jointly sponsored by the
Educational Methods Division and the
Engineering College Administrative Council.)

Printed by Varian Associates

ED000929

GETTING STARTED IN PROGRAMMED INSTRUCTION

by

Robert F. Mager
26 November 1962

Introduction

The aim of this brief article is to help you explore the techniques of programmed instruction; to help you take the first few steps toward evaluating the usefulness of the technique in your own instructional situation. It is not intended to be a thorough-going discussion of the theoretical ramifications and principles of programmed instruction, nor is it intended to make you an expert in the implementation of these techniques.

To begin, let's consider what programmed instruction is. Programmed instruction is a technique that incorporates principles of learning so as to achieve a specific output from a student. It is instructional engineering; it begins with a specified input, and applies a process to achieve a finished product. Programmed instruction is achieved by specifying the desired characteristics of the finished product (that is, the terminal behavior required from the student), by specifying the minimal characteristics of the raw material (prerequisites), and by fabricating a step-by-step process that will mold the raw material into the finished product.

There is nothing mysterious about programmed instruction, and you do not require years of training in psychology to construct a useful program. This is not to say that programming is not hard work, because it is. But whether or not you can become a good programmer depends less upon the depth of your understanding of psychological principles than upon your ability to learn from students.

One of the delightful advantages of programmed instruction is that its use does not necessarily require the destruction or even disruption of an existing course. While programs may be used in different ways at different times, you can have the benefits of programmed instruction without tearing the fabric of your present course. Ultimately the advent of programmed and automated instruction will require some rather serious changes in the way courses are conducted, but the introduction of a modest number of programmed units into a curriculum can be easily accomplished without threatening the structure of the system.

Another advantage of programmed instruction, already implied in the previous paragraph, is that an entire course need not be programmed. Not only can short programmed units be utilized to good advantage, but the short unit is recommended over a completely programmed course.

The following sections provide some brief suggestions about the preparation of programs, and about the use of programs. If you intend to try to prepare a short instructional program--and we strongly urge that you try at least one--then both sections may be useful. If, on the other hand, you intend only to try out someone else's program, then the second section is germane. A short annotated bibliography follows these sections. Many excellent and important articles on programmed instruction have been omitted from this list. The omissions are deliberate. The object of this article and of the bibliography is to suggest ways of getting started; the intent is neither to flood you with all of the literature of this new field nor to imply that you need a good deal more information than you now have in order to get started. You don't. Therefore, this bibliography lists articles that I believe will be immediately useful to you, and I offer my apologies to all the authors of important papers excluded from the list.

Program Preparation

One of the most profound and meaningful statements made about programmed instruction is that the way to learn to write a good program is to write a bad one. You see, a good program is one that works. A program that works is one that teaches to a student what it is intended to teach. The only way to discover whether a program works is to give the student the program and sample his output. While a good programmer, or a psychologist, or a colleague, can make significant inputs relating to the internal structure of your program, only a student can tell you how well it works. If the student fails to learn what the program intended to teach him, the program is a failure, regardless of how well your colleagues like it and regardless of how well you like it. When you design a circuit to achieve a particular output and then discover that the output is deficient, what do you do? Do you "fail" the circuit and complain that it ought to have behaved differently? No, you redesign the circuit. Precisely the same behavior is required in programming. A program is designed, and tried out on one or more students. If it achieves the desired output it is finished; if it does not, it must be revised. Here is an outline of steps you might follow in preparing your first program.

1. To begin, select a single concept or skill that you wish to impart. Do not say to yourself, "I'll program this chapter, or this book, or this course." Begin by trying to program a single concept or idea. One of the most frequent mistakes made by novice programmers is that they bite off more than they can chew, or they attempt to put in program form some fixed body of instructional content. What single idea seems to be most difficult for your students to understand? The answer to that question will give you a lead on what to program.

2. Specify the desired output in as much detail as possible. Here we are talking about instructional objectives, not about content. At this point you must write

down as precisely as you can what the student will be doing when he is demonstrating that he has learned what you want him to know. Note that it is important to describe the student's behavior, rather than some state of his intellect which is not open to inspection. You must indicate how you will tell whether the student has achieved the state or condition you want him to achieve. If you cannot do this, you may have discovered the reason students find the concept difficult to understand. In any case, the more carefully and precisely you prepare your objectives, the easier the remainder of the programming task will be.

3. Prepare a suitable criterion examination based exclusively on the objectives. Unless you can determine whether a student achieved your objectives, you can never be sure whether your program works. Keep in mind that the sole object of this criterion test is to discover whether or not the learner reached the objectives of the program; your purpose is not to build an instrument that will yield a normal distribution of scores, nor to build an instrument that contains no "obvious" questions, nor to build one that contains trick questions that merely test the learner's ability to read and second guess.

4. Describe your raw material. You have to begin someplace, and you have to assume that students begin with some prerequisite body of information or skill. Write out what it is you are assuming a student knows, and keep this document in front of you while you prepare your program draft. Whenever you finish a page of program look it over to see what you have assumed on the part of the student. For example, did you use the word impedance? If you <u>are</u> assuming the student knows the meaning of the term, write this on your prerequisite sheet. If you are <u>not</u> assuming the student knows about impedance, the program must teach it to him. To complete a program frame, will the student be expected to use the table of trigonometric functions or a slide rule? If you are assuming that he has these or other skills, specify them on your prerequisite sheet. If you are not assuming he has these skills, they must be taught.

5. Once you have specified input and desired output, and have developed a way of detecting, if not measuring, the output, you are ready to prepare a program draft. At this point you might want to look over one or two programs that others have prepared. While this is useful, let me urge you not to conclude that the programs you look at represent the only useful programming strategy or the necessary strategy. Actually, this is your chance to innovate. Programming strategies are changing, and while programmed instruction is undoubtedly going to make a significant impact upon the way we teach there is still a good deal of room for improvement and invention. While the last word has been far from said on programming strategy, here are a few programming tips that might ease your way:

    A. The goal is to communicate rather than to use as few words as possible. Feel free to repeat an idea more than once and don't worry

about using a few extra words, providing you are expressing yourself clearly.

B. Whenever you want the learner to attend to something different, tell him so. For example, if you intend to change the subject, preface the discussion with something like, "Now let's consider _____." To put it another way. give the student enough sign posts so that he always knows where he is and where he is going.

C. One of your goals is to remove all obstacles from the path of learning; most of the ideas you have to teach are difficult in and of themselves and require work to learn even under the best of conditions. Therefore, make all illustrations and equations large enough to read easily. Repeat a drawing or photograph as often as necessary so that the learner doesn't have to hunt for it whenever you refer to it. The object is to teach, not to save paper. Use arrows, boxes, color, etc., and anything else you can think of that will call the learner's attention to important elements.

D. When you have finished a draft, review it to see that you have not used vocabulary at or above the level of the subject matter. In other words, don't teach sophomore subject matter with senior vocabulary. Rather teach sophomore subject matter with freshman vocabulary so that words do not get in the student's way. If one of your objectives is to teach him vocabulary then specify this as an objective and do it systematically. Always use a simple word where it will suffice.

E. Avoid foreign words and be very careful of abbreviations.

F. Delete from your draft all of the following words or phrases:

> Obviously...
>
> It clearly follows that...
>
> Of course...
>
> The solution of this problem is left to the diligent student.
>
> Those of you who have had calculus will see that...

G. While it is permissible to allow the student to make an error, it is never permissible to force him into making an error. Where you are asking him to make a response on the basis of insufficient information, give him the alternative of answering, "I don't know." If you are asking him to make an intuitive leap, tell him so; introduce the question by saying something like, "What would you guess the answer to

this is?" or "Have you any idea what would happen if....?" If he comes to the wrong conclusion be sure you do not berate or insult him for not succeeding.

H. Don't worry about writing a program with a low "error rate." Programs wherein each student gets each response correct do not exist and the desirability of such programs is a fiction. Actually, there are several reasons for asking for a student response: to determine whether a communication has been effective is just one of these. Early in the program you will want to include items to screen out learners who do not meet the prerequisites, and if a learner is unqualified for your program he will make errors on these items. You will ask for responses that help facilitate learning, as when you provide practice items. You will want responses to help convince the learner that he has learned, and you will probably do this with test items. You will want to elicit a response that is aimed at directing the learner's attention to specific material, or to keep him involved. You may well ask for a response to determine what information it is appropriate to give him next (e.g., "Now what would you like to know next?").

There are, then, at least four reasons for including items with a high error count; diagnosis, screening, direction of attention, and motivation.

I. Whenever you are stuck, and for whatever reason, you can get unstuck by asking a student to help you. Ask his reaction to the explanation, or to the item you are having trouble with, or ask him to suggest an analogy or example that might help.

J. Your colleague will probably be more trouble than help. While he can be of great help in reviewing the technical accuracy of your material, he will find it impossible to keep from commenting on your choice of subject matter ("They won't <u>really</u> understand it unless you include this and that, etc."), upon the philosophy of programming ("That's not the way I teach it... All you're doing is spoon feeding them... If it ain't hard it ain't learnin'."), or about your style ("You've taken a whole page to say what you could have said in just one sentence."). You will find that your program draft behaves much like an ink blot, and when you ask your colleague to comment on it his comments are likely to tell you more about himself than about your program.

K. To repeat, whenever you are not sure what to do next, ask a student.

L. When you have finished your first draft hold it in your lap and tell it to a student. Ask for responses at the appropriate places, and listen carefully to what he has to say. Whenever he seems puzzled, ask him if he can suggest where and how he is troubled. (This technique also reduces the amount of retyping required.)

M. When you have made the necessary corrections try your next draft on another student. Do to your program whatever it takes to get that student to reach the objectives.

N. Repeat this process until at least three consecutive students do 90 per cent or better on your criterion test. When you have reached this point you are ready to try the program on twenty or thirty students. One of the frequent reasons students fail to achieve 90 per cent or better on criterion tests is the paucity of spaced review items in the early versions of the program.

Using a Program

In general, a program can be introduced into a course in many ways; students can be allowed to work through the material at home, in the library, or in the classroom. However, if you ask students to work through the material outside the classroom and then cover the same material in your lecture, they will likely learn not to bother going through the program; after all, they have other courses to study for. Here are some suggestions:

A. Before giving the student a program explain what a program is and what it hopes to accomplish. Make sure he understands that this material is prepared differently than other materials and that he will have to follow the instructions if he is to get the most from it in the least time.

B. A few of the students may make disparaging comments about the program. Be wary of taking these comments at their face value, since there may be other reasons for apparent dissatisfaction. At one university where the origin of such comments was traced it was found that the students had taken the program home and, because it was novel, had shown it to their wives. The fact that the wives read and could understand the material was quite a blow to the ego of the students. As we all know, Juniors know everything. Students who already know the material covered by a program are also likely to think poorly of it because it looks "too simple." As a matter of fact,

you yourself are likely to make such a comment about a program covering material that is familiar to you.

C. You needn't worry about the fact that a program doesn't look like a very useful reference work; it isn't intended to replace reference works. When students find that they learn from programs they will begin using them as review materials to help them brush up for an exam. The usefulness of a program as study material can be improved by providing a summary page listing the key ideas covered, and including page references to show where these ideas can be found in the program.

D. The few studies that have been done to determine how well students retain what they learn from programs conclude that retention is as good or better than with other methods of instruction. Even on a common sense basis this is reasonable, since you would expect people to retain better that which they understood better in the first place.

E. Avoid using a program that does not have at least objectives and a criterion quiz. The objectives help the student to organize his own efforts, and the criterion quiz helps convince him that he has learned what he is supposed to learn.

Use of Machines

Programmed instruction is a process, a process that occurs when a program is presented to a student according to the conditions intended by the programmer. When a program is presented to a student by means of a programmed text the student must not only attend to the subject matter but must accept the responsibility for presenting the program to himself according to the instructions. Where you can confidently expect the student to follow these instructions and where the instructions are not so complex as to distract the learner from his primary task, a programmed text is adequate.

Automated instruction, on the other hand, is a program that is presented to a learner by a device that accepts the burden of responsibility for presenting the program according to the intended conditions. When instruction is properly automated, the system assumes the burden for presenting appropriate information, for accepting and evaluating student responses, for presenting knowledge of results, and for selecting subsequent material. When your intent is to automate instruction, electro-mechanical equipment is essential.

But while there are many instructional functions that can be accomplished by an automated system, and while there are many instances where a device is indicated, we do not at this time recommend that you concern yourself with hardware, with one exception: Where your intent is to impart skills, mechanization may be indispensable.

Further Information

"It obviously follows that" all of your questions about programmed instruction have not been answered by this brief paper, nor will they all be answered if you read all the available articles on the subject. It may be that one or another of the committee members can answer further questions, and I am sure we will be willing to try.

ANNOTATED BIBLIOGRAPHY

Part I -- Introduction to Programmed Instruction

Goldberg, Irving A. "An Introduction to Programed Instruction." Applied Programed Instruction. (Stuart Margulies and Lewis D. Eigen, Eds.) New York: John Wiley and Sons, Inc., 1962, p. 15-20.

This question and answer article provides brief answers to many of the questions that are frequently asked about programmed instruction.

Lysaught, Jerome P. "Programed Learning and Teaching Machines in Industrial Training." Applied Programed Instruction. (Stuart Margulies and Lewis D. Eigen, Eds.) New York: John Wiley and Sons, Inc., 1962, p. 23-43.

A useful article to anyone learning to program because it describes the history of a programming project. In so doing the problems encountered are identified and workable answers to many important questions are provided.

Rigney, Joseph W., Fry, Edward B. "Current Teaching-Machine Programs and Programming Techniques." Audio-Visual Communication Review, May-June, 1961, Vol. 9, No. 3.

The section entitled "Practical Considerations for Programmers" (pages 14 to 24) offers some useful suggestions from experienced programmers. While reading through this very informative section you should keep in mind that successful programming does not rely on how well you can memorize the jargon of the trade; success in programming depends upon how much learning you can cause your students to demonstrate. In other words, be neither frightened nor impressed by the fact that every kind of item and every kind of frame has been given some sort of name.

Part II -- Principles of Programmed Instruction

Cook, Donald and Mechner, Francis. "Fundamentals of Programed Instruction." Applied Programed Instruction. (Stuart Margulies and Lewis D. Eigen, Eds.) New York: John Wiley and Sons, Inc., 1962, p. 2-14.

The first part of this article (pages 2 through 10) provides a clear and simple discussion of a few of the principles that are applied in the programming of instruction. In addition, guidance is given in the preparation of programs.

Cram, David. Explaining "Teaching Machines" and Programming. San Francisco: Fearon Publishers, 1961.

While there are many books on programming currently on the market, this is one of the few that actually practices what it preaches; the book itself is presented in programmed form. The advantage of working through this book is that the reader actually experiences a linear program while learning about linear programs, and experiences a branching program while learning about branching programs. This should be one of the first items read by one intending to program.

Klaus, David F. "Programming--A Re-emphasis on the Tutorial Approach." Audiovisual Instruction, April, 1961, p. 130-132, 148.

Here is a very good article to read early in the study of programming techniques. Dr. Klaus, a highly experienced man in the field, discusses the characteristics and usefulness of programmed instruction, and provides guidance in the preparation of materials. An important feature of this article is that it makes the distinction between (a) presenting materials to be learned, and (b) making sure that learning will occur.

Glaser, Robert. "Some Research Problems in Automated Instruction: Instructional Programming and Subject-Matter Structure." Programmed Learning and Computer-Based Instruction. (John E. Coulsen, Ed.) New York: John Wiley and Sons, Inc., 1962, p. 67-85.

Several programming strategies are described in this article including the Ruleg System, and should prove very useful in answering questions about the structuring and sequencing of information.

Part III -- Preparation of Instructional Programs

Mager, Robert F. "A Method for Preparing Auto-Instructional Programs." IRE Transactions on Education, 1961, 4, 151-157.

This article lists and briefly describes all the steps that must be accomplished to prepare and test a program. While it does not provide detailed information on how to construct the various kinds of frames, it does provide a checklist of steps involved in the overall programming process.

Mager, Robert F. Preparing Objectives for Programmed Instruction. San Francisco: Fearon Publishers, 1961.

The first step in programming is that of preparing a clear statement of objectives. This semi-programmed book attempts to help the programmer

describe his goals in a way that will allow the statements of objectives to be useful guides to programming.

Stolurow, Lawrence M. Teaching by Machine. U.S. Dept. Health, Education and Welfare, Gov't. Printing Office, 1961.

Chapter Six, Concepts and Techniques, offers specific guidance in the process of programming and briefly summarizes the major features of the programming techniques of ten well-known programmers. For a discussion of programming principles Chapter Five is highly recommended.

Crowder, Norman A. "Automatic Tutoring by Intrinsic Programming." Teaching Machines and Programmed Learning. (A. A. Lumsdaine and Robert Glaser, Eds.) National Education Association, 1960, 286-298.

This article describes intrinsic programming, more popularly referred to as branching programming. Crowder's thesis is that a program that will adjust itself to the needs of each student must be able to do different things for different students. As the various branching patterns described in this article are studied, the reader should keep in mind that an infinite number of branching patterns exist and that he can use whatever procedure appears most appropriate, and that the ultimate usefulness of the program will be determined by its teaching power rather than by its conformity to some arbitrary set of rules. This article, along with Cram's book, provides an excellent introduction to intrinsic programming.

"The Multiple Choice Frame." Programed Instruction, Feb., 1962, Vol. 1, Bull. No. 4, p. 4.

The fact that the world abounds in poorly written multiple choice items should not lead one to conclude that the multiple choice item per se is worthless. This one page article offers a useful discussion of the multiple choice item and suggests one or two of the several uses for such items.

"Tips for Programmers." Auto-Instructional Devices for Education and Training, March, 1962, p. 108.

This article describes a technique aimed at insuring the desired response on the part of the student in a linear program. It describes the technique and advantages of restricting the possible responses the student might make by including the first letter of the desired response in the blank, by using a broken underline to indicate the number of letters in the desired response and by regulating the length of the blank. While this article is of practical value to the novice programmer, it should be read after three or four students have tested your first draft of your first program.

"Faulty Frames." Programed Instruction, Feb., 1962, Vol. 1, Bull. No. 4, p. 6.

This two page article provides examples of good and faulty linear frames and discusses the importance of the response asked for from the student.

Homme, Lloyd E. and Glaser, R. B. "Problems in Programming Verbal Learning Sequences." Teaching Machines and Programmed Learning. (A. A. Lumsdaine and Robert Glaser, Eds.) National Education Association, 1960, p. 486-496.

This paper describes one of the variations of linear programming, the Ruleg System. It is recommended for reading only after you have written and tested at least one program draft. The first part of this paper deals with some major issues and classes of variables in the development of programmed materials. Its major contribution is that it describes and illustrates the characteristics of several instructional devices and discusses the integration of program and device.

Part IV -- Periodicals

Two of the periodicals in the field contain information of value to the new programmer or program user.

Auto-Instructional Devices (AID). Published monthly by INRAD Educational and Training Methods Division, P. O. Box 4456, Lubbock, Texas. ($5.00 one year.)

In addition to a current bibliography of books and articles in the field, this periodical contains news about new devices and an occasional programming tip. It is worth reading.

Programed Instruction. Published bimonthly by The Center for Programed Instruction, Inc., 365 West End Avenue, New York 24, New York. (No charge.)

This periodical contains reviews of recent books and articles, comments from well-known figures in the field, and some instruction in programming strategy. It is recommended reading even though limited almost entirely to concern for the linear program in paper format.

PLEASE DETACH THIS FORM AND MAIL TO:

Arnold Roe
Chairman
ASEE Committee on Programed Instruction
Department of Engineering
University of California
Los Angeles 24, California

On receipt of this form by the Committee chairman, your name will be entered on the Committee's mailing list so that you may receive supplements to this report and other information on program instruction activities.

Please complete the following:

Full Name _____

Position _____

Affiliation _____

Mailing Address _____

Have you written (Yes☐ No☐), or do you plan to write (Yes☐ No☐) programed material pertinent to engineering education?

In what subject areas? _____

Do you want a listing of other people who have written or are preparing programs in your subject area (Yes☐ No☐)?

Would you like to have your programs reviewed (Yes☐ No☐) or tested (Yes☐ No☐) by people in other engineering schools or industries?

Date _____